Hw

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/823,182	03/29/2001	Daniel M. Lavery	42390.P10809	9287

7590 06/29/2005

R. Alan Burnett
BLAKELY, SOKOLOFF, TAYLOR & ZARMAN LLP
Seventh Floor
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026

EXAMINER	
FOWLKES, ANDRE R	
ART UNIT	**PAPER NUMBER**
2192	

DATE MAILED: 06/29/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

[illegible]

Office Action Summary	**Application No.** 09/823,182	**Applicant(s)** LAVERY ET AL.	
	Examiner Andre R. Fowlkes	**Art Unit** 2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 April 2005*.
2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.
3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-3 and 5-30* is/are pending in the application.
4a) Of the above claim(s) _____ is/are withdrawn from consideration.
5)☐ Claim(s) _____ is/are allowed.
6)☒ Claim(s) *1-3 and 5-30* is/are rejected.
7)☐ Claim(s) _____ is/are objected to.
8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.
10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
a)☐ All b)☐ Some * c)☐ None of:
1.☐ Certified copies of the priority documents have been received.
2.☐ Certified copies of the priority documents have been received in Application No. _____.
3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

[illegible]

DETAILED ACTION

1. This action is in response to the RCE amendment, filed 4/12/05.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-3, 5-7, 9-18, 20-25 and 27-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Dehnert et al. (Dehnert), U.S. Patent No. 6,059,839 in view of Novak et al. (Novak), "A simple mechanism for improving the accuracy and efficiency of instruction-level disambiguation".

As per claim 1, Dehnert discloses a method for performing memory disambiguation in a compiler, comprising:

- determining memory objects corresponding to memory references in one or more source files being compiled (col. 3 lines 60, "determining address data of memory references in software code"),

- creating a unique memory disambiguation file **for each of the memory references, each unique memory disambiguation** file **identifying information particular to the memory reference it is associated with so as to preserve high-**

level and intermediate-level semantic information (Fig. 2, the Intermediate Representation File (IRF) contains information (i.e. unique memory disambiguation files) particular to the memory reference it is associated with, as disclosed in steps 202, 203, 206 and 207, and associated text, such as (col. 5 line 11 – col. 6 line 49), "The inter procedural analysis phase (IPA) analyzes the information propagated by the IRF, step 206. IPA analyses the variables globally across all procedures... It pulls all variable data from the IRF. The IPA determines which variables must be treated as address-saved, inhibiting optimization as described above. The output of the IPA phase is precise address-taken information about all global variables and formal parameter variables. In the global optimization phase, the back end of the compiler uses the precise address-taken information from the summary tables generated by IPL and IPA (i.e. from the IRF) to optimize the code as thoroughly as possible, in step 207. The precise address-taken information generated by the IPL phase and the IPA phase of the present invention have provided for the full disambiguation of the memory references (variables, parameters, and the like). The references which cannot be optimized without inducing errors are particularly identified", and 4:5-26, "Next, the inter-procedural local analysis phase of the compiler (IPL) analyzes all of the procedures and all of the local data references inside the procedures for each variable. The IPL determines what kind of references occur for each variable within the procedure. The IPL (uniquely) summarizes all these local cases (into unique memory disambiguation files). The inter-procedural analysis phase of the compiler (IPA) analyzes the variables globally across all procedures. IPA determines which variables have had their addresses taken and used in ways which inhibit optimization. IPA (uniquely) summarizes all these global cases. The summaries provide precise address-taken information about all variables even when the variables are global or otherwise externally visible. The precise address-taken information generated by the compiler of the present invention thus aids the disambiguation of memory references. Those references which cannot be safely optimized are particularly identified (using the unique memory disambiguation file)").

- **creating a symbolic memory reference** file **associated with each memory disambiguation** file**, including information on whether the memory reference is indirect or direct and access to symbol table information for a pointer to the memory object for indirect references or the memory object for direct references** (Fig. 2, the IRF contains information, including whether the memory reference is indirect or direct and access to the information contained in a symbol table, as disclosed in item 205, and associated text, (e.g. col. 5 line 11 – col. 6 line 49)"),

- **determining if potentially dependent memory references are dependent or independent based on information contained in the disambiguation** files **for those memory references and their associated symbolic memory reference** files (col. 4 lines 1-3, "The compiler analyzes the operations the program actually performs to determine which (memory references are independent and) can be optimized").

Dehnert doesn't explicitly disclose that **the unique disambiguation token comprises a data structure including a plurality of links to data objects in which disambiguation information are stored.**

However, Novak, in an analogous environment, discloses that **each unique memory disambiguation token comprises a data structure including a plurality of links to data objects in which disambiguation information are stored** (p. 293, ¶ 3 lines 1-4, "the data structure created by the compiler during parsing and source-level analysis—which maintain, for example, defuse and alias information (i.e. disambiguation info)", and the token is described on p. 293, ¶ 5 line 4 – p. 294, ¶ 1 line 2, "allowing the front-end to communicate higher level, memory reference information to the back-end by associating each reference with a portion of a hierarchical decomposition of the program's address space").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Novak into the system of Dehnert to have each unique memory disambiguation token comprise a data structure including links to data objects in which disambiguation information is stored. The modification would have been obvious because one of ordinary skill in the art would want to allow the front end of a compiler to communicate higher level memory reference information to the back-end in order to allow for increased efficiency and accuracy of optimization of the code (Novak, p. 293 ¶ 5 line 1 – p. 294 ¶ 1 line 2).

As per claim 2, the rejection of claim 1 is incorporated and further, Dehnert discloses **determining if the memory references are redundant based on information contained in the unique memory disambiguation** files **for those memory references and their associated symbolic memory reference** files (col. 2

lines 2-4, "the compiler may (use the information in the IRF (i.e. the memory disambiguation file) to) ... place code in locations such that a particular calculation called for in the code can be performed once, instead of (many times)").

As per claim 3, the rejection of claim 1 is incorporated and further, Dehnert discloses **determining a relative difference in starting addresses for two memory references** (col. 2 lines 15, "The compiler must make sure not to move certain units of code, especially memory references, past other units of code which refer to the same memory location", and the relative difference between starting addresses for two memory locations must be determined before the compiler moves code units).

As per claim 5, the rejection of claim 1 is incorporated and further, Dehnert discloses that **the data structure is embedded in memory reference operators of an intermediate language produced during the compilation of the one or more source files** (col. 6 lines 20-21, "The IRF (i.e. intermediate language produced during compilation) includes the additional data gathered about the variables").

As per claim 6, the rejection of claim 1 is incorporated and further, Dehnert discloses that **the unique memory disambiguation** file **associated with the memory object includes** access capability **that is used to access data dependence information** (col. 4 lines 1-3, "The compiler analyzes the operations the program actually performs to determine (data dependence information and decide which memory references) can be optimized").

As per claim 7, the rejection of claim 1 is incorporated and further, Dehnert doesn't explicitly disclose that **the unique memory disambiguation token contains a link to address base and offset information for the memory reference that is used for low-level disambiguation**.

However, Novak, in an analogous environment, discloses that **the disambiguation token contains a link to address base and offset information for the memory reference that is used for low-level disambiguation** (p. 293, ¶ 5 line 4 – p. 294, ¶ 1 line 2, "allowing the front-end to communicate higher level, memory reference information to the back-end by associating each reference with a portion of a hierarchical decomposition of the program's address space (i.e. address base and offset information)").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Novak into the system of Dehnert to have the disambiguation token contains a link to address base and offset information for the memory reference that is used for low-level disambiguation. The modification would have been obvious because one of ordinary skill in the art would want to use as much information as possible in order to maximize the possible code optimizations (Novak, p. 293 ¶ 5 line 1 – p. 294 ¶ 1 line 2).

As per claim 9, the rejection of claim 1 is incorporated and further, Dehnert discloses **using information identified by disambiguation** files **to determine sets of local memory objects that are not referenced after they are modified** (col. 4 lines 8-11, "(The system) determines what kind of references occur for each variable ... (the system, then) summarizes all these local cases").

As per claim 10, the rejection of claim 1 is incorporated and further, Dehnert discloses **determining if two memory references access overlapping memory locations based on information contained in the disambiguation** files **for those memory references and their associated symbolic memory reference** files (col. 5 lines 37-40, "The compiler of the present invention determines the nature of the operations the program actually performs with the addresses. The ultimate objective is determining which of the (memory references access overlapping memory locations).").

As per claim 11, the rejection of claim 10 is incorporated and further, Dehnert discloses **determining particularities of an overlap between two overlapping memory references** (col. 5 lines 56-59, "(The system) analyzes all of the procedures and all of the data references inside the procedures)...(The system, then) compares and categorizes all local references to the variables").

As per claim 12, the rejection of claim 1 is incorporated and further, Dehnert discloses:

- **determining the functions executed corresponding to function calls in the one or more source files being compiled** (Fig 2 item 204, "Analyze local procedures and data calls", and associated text, (e.g. col. 5 line 11 – col. 6 line 49)),

-**creating a disambiguation** file **for each function call, each disambiguation** file **identifying information particular to the function call it is associated with so as to preserve high-level and intermediate level semantic information** (Fig 3 items 307, "Analyze attributes for all procedure calls", and 315, "Summarize procedure call variable information (in a file)", and associated text, (e.g. col. 6 line 50 – col. 8 line 12)),

-**creating a symbolic function call** file **associated with each disambiguation** file, **including information on whether the function call is indirect or direct and access to symbol table information for the pointer or function respectively** (Fig 3 items 308, "(determine if procedure calls are) direct or indirect", and 315, "Summarize procedure call variable information (in a file)", and associated text, (e.g. col. 6 line 50 – col. 8 line 12)),

- **determining if potentially dependent calls and memory references are dependent or independent for the function calls based on information contained in the disambiguation** files **for the calls and memory references, their associated symbolic** file, **an analysis of each function to determine the set of memory locations modified or referenced by the function** (col. 4 lines 1-3, "The compiler analyzes the operations the program actually performs to determine which (memory references are independent and) can be optimized").

As per claim 13, the rejection of claim 1 is incorporated and further, Dehnert doesn't explicitly disclose that **the disambiguation token contains a link to type information associated with the memory reference**.

However, Novak, in an analogous environment, discloses that **the disambiguation token contains a link to type information associated with the memory reference** (p. 297 ¶ 1 lines 3-7"(the disambiguation token contains type information to allow the) dependence analysis tool (to ensure that) ... pointer variables will be associated only with paths for which the appropriate 'type' node is present").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Novak into the system of Dehnert to have the disambiguation token contain a link to type information associated with the memory reference. The modification would have been obvious because one of ordinary skill in that art would have wanted use type information to improve the disambiguation of pointer references (Novak, p. 297 ¶ 1 lines 7-9).

As per claim 14, the rejection of claim 1 is incorporated and further, Dehnert discloses **the disambiguation** file **for an indirect memory reference contains a link to a set of memory objects accessible via the pointer as determined by points-to analysis** (Fig. 5, item 502, and associated text (e.g. col. 8 lines 36-47) show the summary of the disambiguation file for an indirect memory reference).

As per claim 15, the rejection of claim 1 is incorporated and further, Dehnert doesn't explicitly disclose **using the unique memory disambiguation token and the symbolic memory reference** file **as an interface or means of communication between various software components of a disambiguator that performs memory disambiguation functions and clients of the disambiguator**.

However, Novak, in an analogous environment, discloses **using the unique memory disambiguation token and the symbolic memory reference** file **as an interface or means of communication between various software components of a disambiguator that performs memory disambiguation functions and clients of the disambiguator** (p. 293, ¶ 5 line 4 – p. 294, ¶ 1 line 2, "allowing the front-end (software component) to communicate higher level, memory reference information to the back-end (software component) by associating each reference with a portion of a hierarchical decomposition of the program's address space").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Novak into the system of Dehnert to use the unique memory disambiguation token and the symbolic memory reference file as an interface or means of communication between various software components of a disambiguator that performs memory disambiguation functions and clients of the disambiguator. The modification would have been obvious because one of ordinary skill in the art would want to use as much information as possible in order to maximize the possible code optimizations at each level of compilation (Novak, p. 293 ¶ 5 line 1 – p. 294 ¶ 1 line 2).

As per claims 16-18 and 20-22, this is a system version of the claimed method discussed above, in claims 1- 3 and 9-12, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see Dehnert's apparatus and method for compiler identification of address data (col. 2:2-8:47).

As per claims 23-25 and 27-30, this is an article of manufacture version of the claimed method discussed above, in claims 1-3, 9-12 and 15, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see Dehnert's apparatus and method for compiler identification of address data (col. 2:2-8:47).

4. Claims 8, 19, and 26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Dehnert et al. (Dehnert), U.S. Patent No. 6,059,839 in view of Rountev et al. (Rountev), "Off-line variable substitution for scaling points-to analysis".

As per claim 8, the rejection of claim 1 is incorporated and further, Dehnert doesn't explicitly disclose **substituting a direct memory reference for an indirect memory reference.**

However, Rountev, in an analogous environment, discloses **substituting a direct memory reference for an indirect memory reference** (p. 2, col. L lines 30-33, "we propose off-line variable substitution").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Rountev into the system of Dehnert to have substitution of direct memory references for indirect memory references. The modification would have been obvious because one of ordinary skill in the art would want to reduce the cost of points-to analysis (p. 2, col. L lines 30-33).

As per claims 19 and 26, Dehnert in combination with Rountev also discloses such claimed limitations as addressed in claim 8 above.

Response to Arguments

5. Applicants arguments have been considered but they are not persuasive.

In the remarks, the applicant has argued substantially that:

1) There is no motivation or suggestion to combine Dehnert and Novak, at p. 11:1-9.

Examiner's response:

1) In response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in

the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988)and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, one of ordinary skill in the art would have been motivated to allow the front end of a compiler to communicate higher level memory reference information to the back-end in order to allow for increased efficiency and accuracy of optimization of the code, (Novak, p. 293 ¶ 5 line 1 – p. 294 ¶ 1 line 2).

2) Dehnert's IRF contains data about memory <u>locations</u>, in contrast to applicant's invention which contains data about memory <u>references</u>, at p. 11:14-12:20.

Examiner's response:

2) The examiner disagrees with applicant's characterization of the applied art. Dehnert's IRF clearly contains data about memory references, as disclosed by col. 5:53-67, "the IPL analyzes the IRF and analyzes all of the procedures contained therein. The IPL analyzes all of the procedures and <u>all of the data references</u> inside the procedures (that are inside the IRF)".

In the remarks, the applicant has argued substantially that:

3) Novak does not teach a memory disambiguation token identifying information particular to the memory reference it is associated with so as to preserve high-level and intermediate level semantic information, at p. 13:5-10.

Examiner's response:

3) The examiner disagrees with applicant's characterization of the applied art. Novak does disclose a memory disambiguation token, as disclosed in the above art rejection. Additionally, Novak was not cited by the examiner to teach identifying information particular to the memory reference it is associated with so as to preserve high-level and intermediate level semantic information. Dehnert discloses this feature as addressed in the art rejection, above. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.,* 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

Conclusion

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andre R. Fowlkes whose telephone number is (571) 272-3697. The examiner can normally be reached on Monday - Friday, 8:00am-4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571)272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

ARF

[illegible]

TUAN DAM
SUPERVISORY PATENT EXAMINER